The PHENIX High Voltage Control System

(http://www.phenix.bnl.gov/ phoncs/oncs/Anc_sys/hvmanual.ps)

July 16, 2000

Lars Ewell

ewell@bnl.gov

Johann Heuser

Johann.Heuser@sunysb.edu

PHENIX High Voltage Control

Outline

Introduction

The intent of this document is to describe the detailed operation of the high voltage control system for the PHENIX detector at Brookhaven National Laboratory. The high voltage control system makes use of the Experimental Physics and Industrial Control System (EPICS) software package. First discussed will be the computers that act as channel access servers and clients. Following this a short overview of EPICS and in particular, EPICSB will be given followed by a description of the MEDM GUI that will be used and finally a brief description of how the Objectivity database will be utilized.

It should be noted that there has already been a substantial amount of documentation written regarding this control system. The reader is encouraged to peruse these existing documents on the web and elsewhere (e.g. the ONCS home page (http://www.phenix.bnl.gov/ phoncs/oncs/oncshome.html) and then the *Ancillary Systems* link) . References are noted throughout the text as well as in the bibliography.

Computers

Phoncs0

Phoncs0 is the host computer on which the channel access client operates as well as where all of the EPICS software code is compiled to be loaded onto various Input Output Controllers (IOCs). It is an Ultra Sparc Enterprise 3000 and runs Solaris 2.6 (see http://www.phenix.bnl.gov/ kehayias/phoncs.html). The directory structure of the relevant area is depicted in Figure 1.

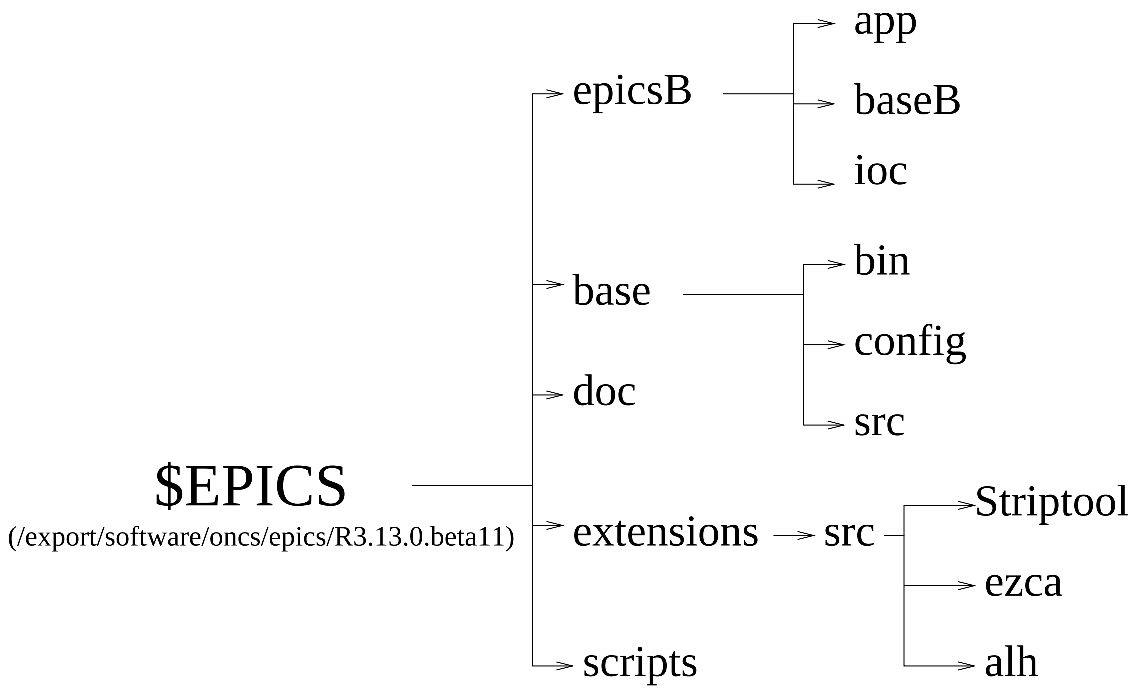

Figure 1: Directory structure on Phoncs0

Note that in the $EPICS/scripts directory is a setup_epics script that sets various environment variables needed to work in this area.

Phoncs5

Phoncs5 is the host computer on which the main Motif Editor and Display Manager (MEDM (see MEDM chapter)) will be displayed during normal PHENIX running. It is a SPARCstation 5 and is located in the southwest corner of the counting house.

IOC

The IOCs that function as the channel access servers are Motorola MVME167s. They use the Motorola 68040 processor and have memory extension from the standard 4M to 16M. The operating system on the IOCs is VxWorks 5.3.1. It is important that the IOC be set up properly. When booting up the IOC, the setup screen is one of the first things that shows on the screen. An effective way to view the setup screen of the IOC is to connect the **Serial Port 1/ Console** connector located on the IOC (either on the front panel or on the back of the VME rack near the transition card) to a VT220 terminal. In order to change the IOC parameters, it's necessary that the VT220 terminal be setup correctly. The setup key (F3) shows the setup for the VT220. In particular, under the "General" Menu the "New Line" entry should be set to "No New Line" by toggling using the enter key on the numeric keypad.

Once the VT220 is hooked up, the IOC can be rebooted by pushing the reset button on the front panel. The VT220 should then show the message "Press any key to stop autoboot....". To view and/or change the configuration parameter, first stop the autoboot by pressing a key. Then, by typing " p " at the [VxWorks Boot]: prompt, the configuration parameters are printed on the screen. For iocondev5, the configuration parameters are shown in Table 1.

Note that the IP address for the IOC is the host on the backplane. To view the IP address for different IOCs, see
http://www.rhic.bnl.gov/phenix/project_info/electronics/maps/phenix_network.txt.

The IOC occupies the leftmost (or bottom when using a 'mini'-crate on it's side) slot of the VME crate. The memory extension to the IOC enlarges it from one to two slots. Located in the slot next to the IOC is the VME bus COMPCONTROL CC121 arcnet card that allows for the IOC to communicate with the LeCroy 1458 mainframes.

Note that recently (3/25/00) there has been established direct serial connection from the transition card on the rear of the IOCs to the terminal server, phoncs8. This allows for direct connection to the IOCs by utilizing the shell script 'ppcserial'. To establish connection, type 'ppcserial iocondev5' and a direct serial connection should be established.

Table 1: IOC configuration parameters

boot device	: ei
processor num	: 0
host name	: phoncs0.phenix
file name	: /home/phoncs/config/vxboot/vxWorksT101_68k167
inet on ethernet (e)	: 130.199.98.105:ffff0000
host on backplane (b)	: 130.199.98.105
host on inet (h)	: 130.199.98.60
gateway inet (g)	: 130.199.98.24
user (u)	: ftpuser
flags (f)	: 0x0
target name (tn)	: iocondev5
startup script (s)	: /home/phoncs/config/vxboot/iocondev5.phenix.68k

Once the IOC has been booted and configured correctly, it is more convenient to use an "xterm" to communicate with it over ethernet. This allows for scroll bar operation so that there exists a short record of what happens for reference. To do this, the remote shell "rsh" command can be used. Type "rsh iocondev5" to connect. "Control-x" then reboots the IOC from the xterm window.

High Voltage Power Supplies

LeCroy 1458 Mainframe

The main high voltage power supply to be used in PHENIX is the Lecroy 1458 "Mainframe" (MF). This has the capability of utlizing up to sixteen different modules and has an onboard 486 processor for communication as well as for performing simple operations. The reader is referred to the LeCroy Research Systems User's Guide V3.04 for detailed information regarding this supply. There are several copies of this manual for perusal in the PHENIX counting house.

As referred to above, the arcnet protocol is used to communicate with the 1458 in regular operation. However, it's also possible that one may want to bypass arcnet and the IOC altogether when communicating with the MF. This is possible using the serial communication port. One can use, e.g., the hyperterminal available on a PC running windows NT in conjunction with the serial port to connect to and communicate with the 1458 mainframe. Details of this are included in the LeCroy manual.

In addition to the usual EPICS communication with the MF, there exists another 'bare bones' communication method useful for troubleshooting. This is the 'hvcontrol' system developed by Hua. There is extensive documentation in the $EPICS/epicsB/app/hvcontrol area (see Figure 1).

Macro to Enable Interlock

On the front panel of the 1458 Mainframe are located three BNC connections labeled "Status", "Macro" and "Interlock". If an interlock signal (e.g. overtemperature or rack door open) is desired, the signal should be connected to the "Macro" connector. In addition to this, a macro that is internal to the mainframe needs to be set.

The setting of the macro is best achieved via the *hvcontrol* program referred to above. To proceed with this, first reboot the desired IOC by issuing the *control x* command from the IOC window. When the bootscript stops in the $IOC/iocondev*x* directory, cd to the hvcontrol directory via
cd "/export/software/oncs/epics/R3.13.0.beta11/epicsB/app/hvcontrol/ioc " (note the double quotes around the directory required by VxWorks). Then
< *load_hvcontrol_iocondev***x**. If the bootscript gets excecuted properly, the last line should look like:

——-

hello, arcnet is initialized arcnet node id read from the board itself = 2.

——-

Then type *hvcontrol* to start the program. A list of the mainframes on the network, with their status (HVON/HVOFF) should be echoed to the screen followed by the question "which mainframe? ". Enter the desired mainframe (note: the mainframe number is in decimal format in distinction from the hexadecimal format of the Excel spreadsheet referred to below). Then at the "mainf: XX ->" prompt, type */ sysdef* (note the space between the / and sysdef). The sysdef number should be echoed back to screen. The default number is 000B. To enable the macro for the interlock, the sysdef number needs to be 008B. To change the number, type */ sysdef 008B*. Then to check to see that the new number has been accepted, again type */ sysdef*. If the new number 008B is then echoed to the screen, the macro is now enabled and the interlock can work. For further information, the reader is referred to the LeCroy HV 1458 Mainframe manual.

1461, 1469, and 1471 Modules

In Table 2 we see the High Voltage Card Comparison Matrix for the different types of HV modules. All three types of modules (1461, 1469 and 1471) will be used in the PHENIX detector. There is an Excel spreadsheet detailing the HV MF populations along with their arcnet address located at
http://www.phenix.bnl.gov/phenix/project_info/electronics/maps/HV_ARCNET.xls.
In addition, there is a page devoted to some of the problems encountered with individual modules during the engineering run located at
http://www.phenix.bnl.gov/ phoncs/oncs/Anc_sys/epicsTrouble.html.

Table 2: High Voltage Card Comparison Matrix
1461, 1469, 1471 Modules

Specifications	Model 1461	Model 1469	Model 1471
Description	12 ch., Max. 3kV, Max. 2.5mA/ch.	24 ch., Max. 3.5 kV, Max. 98μ A/ch.	8 ch., Max. 6 kV, Max. 200 μA/ch
Channels	12 fully independent	24, programmable in groups of 8	8 fully independent
Voltage	Programmable 0 to 3 kV	Programmable 0 to 3.5 kV	Programmable 0 to 6 kV
Voltage Polarity	1461N for neg. 1461P for pos.	1469N for neg. 1469P for pos.	1471N for neg. 1471P for pos.
Voltage Set Resolution	$<$ 1 V (750 mV nominal)	$<$ 1 V (900 mV)	$<$ 1 V (500 mV nominal)
Voltage Output Matching	N/A	$<$ 500 mV variation within group	
Voltage Output Accuracy @ 25^o	$\pm$ (0.1% of setting + 1.5V) from 5% to 100% of full scale. (Below 5% a min. load may be necessary)	$\pm$ (0.1% of setting + 1.5V)	$\pm$ (0.1% of setting + 3 V) from 5% to 100% of full scale. (Below 5% a min. load may be necessary)
Temperature Stability	$<$ 100 ppm/ ^{o} C	$<$ 100 ppm/ ^{o}C	$<$ 100 ppm/ ^{o} C
Voltage Repeatability	$< \pm$ 0.5 V at constant load, and temp.	$< \pm$ 0.5 V at constant load, and temp.	$< \pm$ 3.0 V at constant load, and temp.
Voltage Output Ripple	$<$ 100 mV p-p, $<$ 50 mV p-p for $<$ 1 mA or freq. $>$ 25 kHz	$<$ 50 mV p-p, $<$ 25 mV p-p for freq. $>$ 25 kHz	$<$ 50 mV p-p, $<$ 10 mV p-p for freq. $>$ 1 kHz
Voltage Measurement Resolution	$<$ 1 V (750 mV nominal)	$<$ 1 V (900 mV)	$<$ 1 V (500 mV)
Voltage Measurement Accuracy @ 25^o	$\pm$ (0.1% of reading +1.5V	$\pm$ (0.1% of reading +1.5V	$\pm$ (0.1% of reading +3V
Voltage Ramp Rates	Programmable/ch. separate ramp up & down rates (nominally 50 to 2000 V/s in 50 V steps)	Programmable/ch. separate ramp up & down rates (nominally 1 to 500 V/s in 1 V steps)	Programmable/ch. separate ramp up & down rates (nominally 1 to 500 V/s in 1 V steps)
Output Current Capability	$>$ 2.5 mA (2.8 - 3kV), $>$ 1.0mA (0 - 1kV), linear derate from 1 to 2.8 kV	98μA; (300μA charging ave./ch./ within bulk	200μ/ch.

Table 2 (con't)

Specifications	Model 1461	Model 1469	Model 1471
Current Measurement Resolution	$< 1\mu$A	26nA	15nA
Current Measurement Accuracy	$\pm$ (2% of reading + 15 μA)	$\pm$ (2% of reading + 100 nA)	$\pm$ (2% of reading + 50 nA)
Current Trip	Programmable/ch. 1 μA resolution (from 50 to 2550 μA)	Programmable/ch. 26nA resolution (from 5 to 98 μA/ch.) relay disconnect 2.5mA bulk trip	Programmable/ch 15nA resolution (from 1 to 200 μA; 15 nA resolution
Current Trip Detect Time	< 200 msec	< 10 msec	< 10 msec (normally 2msec)
24 V Power Requirement	171 mA supply per mA output + 38 mA supply per channel. Multiple modules can exceed the power supplied by the standard crate.	< 208mA	330 mA supply per mA output + 56 mA supply per channel.
Hardware	One potentiometer & 1000:1 test point	One potentiometer & 1000:1 test point	One potentiometer & 1000:1 test point
HV ON LED	One; steady on for all channels stable HV, flash for any channel changing output or trip	One; steady on for all channels stable HV, flash for any channel changing output or trip	One; steady on for stable HV, all channels flash for any channel changing output or trip
Dimensions	6 U (10.3" high x 14.6 " deep x 1" wide; Eurocard C size)	6 U (10.3" high x 15.0 " deep x 1" wide; Eurocard C size)	6 U (10.3" high x 14.6 " deep x 1" wide; Eurocard C size)

There are, in addition to the LeCroy 1458, two other types of high voltage power supplies that will be used in the PHENIX detector.

Caen

The Time Of Flight (TOF) subsystem will use a Caen power supply. As of the time of this report, Akio Kiyomichi (kiyo@bnl.gov) is a person familiar with the TOF system.

HiVoc

Finally, there is a custom made high voltage supply called HIVOC to be used with the lead glass detector. This was used in CERN experiment WA98 and some information on it can be obtained from Martin Purschke's web page
(see http://www.cern.ch/WA98/PUBLICATIONS/LeCroyProc.ps).
There is some code located on phoncs0 at /home/phoncs/mtest/hivoc as of the time of this printing.

PHENIX EPICS

The Experimental Physics and Industrial Controls Systems (EPICS) was chosen to provide for control of PHENIX high voltage. There exists extensive documentation on the web and elsewhere. The reader is encouraged to peruse the existing PHENIX web pages (see http://www.phenix.bnl.gov/ phoncs/oncs/Anc_sys/anc_home.html) and links therein.

The version of EPICS used for PHENIX is R.3.13.0.beta11. The PHENIX EPICS implementation is based on an similar system developed at Hall B of Jefferson Laboratory. For a detailed description of this, the reader is referred to the well written master's thesis of either Thierry Auger (South Carolina, 1996) or Marc Winoc Swynghedauw (South Carolina, 1996). Copies of these are available in the PHENIX counting house.

In Figure 2 we see a diagram of the PHENIX EPICS high voltage system. In the MVME167, we see a portion of EPICS database that consists of two records: a hiv (high voltage) record with name "HV_..." and an ai (analog input) record with name "B/HV...". The EPICS database is of primary importance in the control system.

The EPICS database consists of different 'records', which in turn may correspond to an individual high voltage channel. Each of the records has a unique name (see http://www.phenix.bnl.gov/ phoncs/oncs/conventions/convention_home.html for the naming convention for HV channels). To view the records that exist in the database, type "dbl" (for database list) at the "iocondev3>" prompt. An individual record has many different "fields" that may represent an alarm level, scan rate, voltage level etc. For a list of commands that can be issued from the "iocondev3>" prompt once the database has been loaded, see chapter 7 of the *EPICS IOC Application Developer's Guide* a copy of which is located in $EPICS/doc (see Figure 1).

The EPICS database is created by issuing a 'make -s iocondev(x)' (where x is 3, 4 or 5) command in the $EPICS/epicsB/app/db directory. When this command is issued, the perl script 'hv2db' is invoked with the proper '.dat' file as an argument. The output of this is the EPICS database with a .db suffix. This process will be discussed more thoroughly in the MEDM GUI chapter.

EPICSB

In the 'final' HV configuration, we expect there to be approximately 3,500 'records' in the EPICS database. The vast majority of these will be the 'high voltage' custom records of EPICSB. This hiv record was made specifically for the LeCroy 1458 down

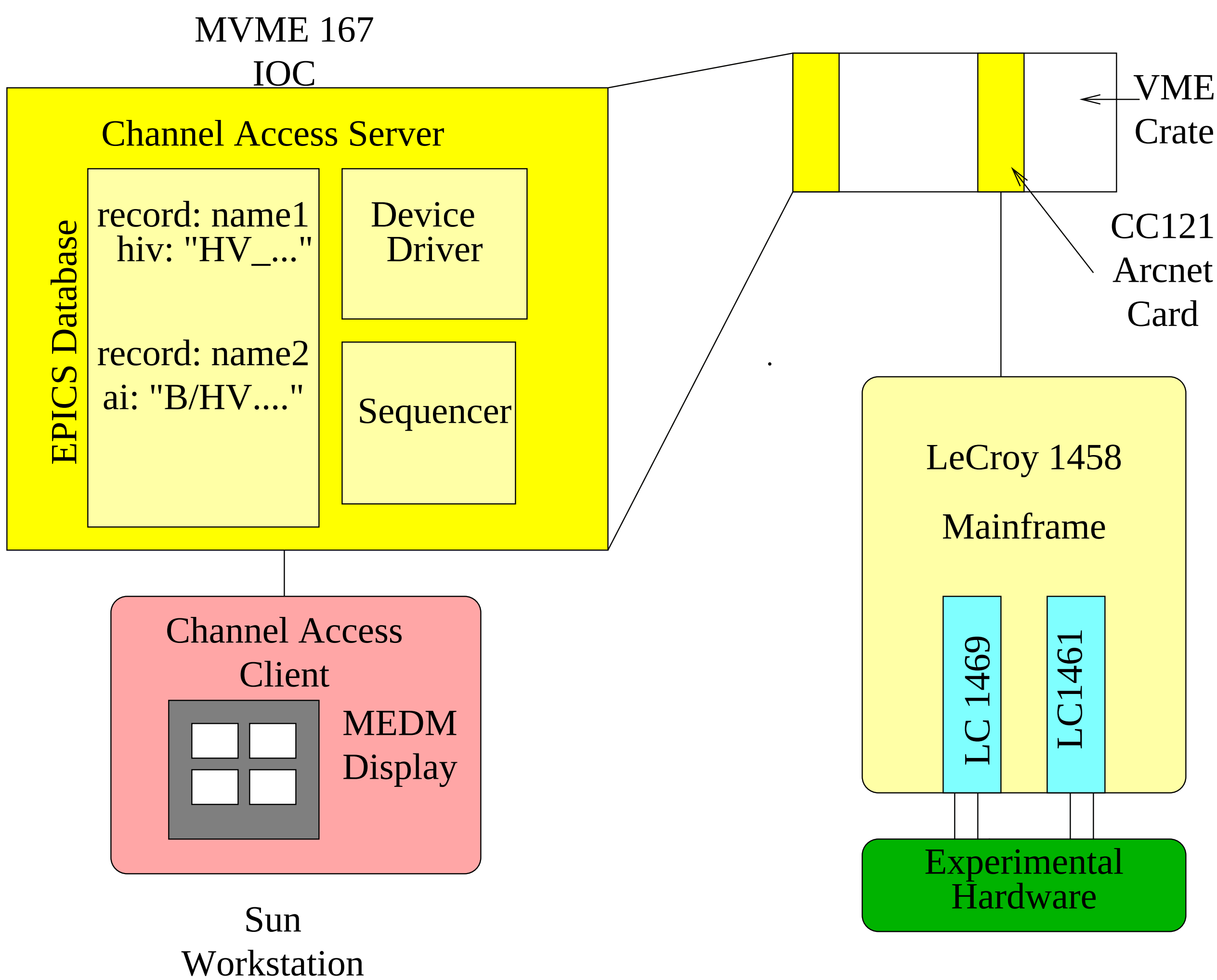

Figure 2: EPICS High Voltage System

at the Jefferson Laboratory for the CLAS detector in Hall B. The main feature that distinguishes it from an ordinary EPICS high voltage record is that it has multiple 'VAL' fields to read and write to. Due to this, at the current time the driver code needs to be compiled in it's own separate directory (see Figure 1) and is compatible only with EPICS version 3.13.0.beta11.

The alarm handler is an EPICS extension that provides for the ability to set alarms on individual channels so that the user is notified when a channel trips is disabled, or exceeds limits on voltage and/or current. In the $EPICS/epicsB/app/medm/alh directory is located the alarm configuration file with a suffix '.alhConfig'. This file is created when a 'make -s gui' is done in the $EPICS/epicsB/app/medm directory. To start the alarm handler, type 'alh phhvMaster.alhConfig' at the Unix prompt on phoncs5. A widget similar to Figure 3 should pop up.

Figure 3: EPICS Alarm Handler Button

Under normal operation, this should be grey colored and silent. If an alarm condition occurs, this button will turn either white, yellow or red and will flash. White indicates that communication has been lost between the IOC and hardware. Yellow indicates that either a "low" or "high" limit has been exceeded, or a channel has been disabled. Red indicates that a either a "low low" or "high high" limit has been exceeded, or a channel has tripped. If the flashing button is pushed (left mouse click), a panel similar to Figure 4a should pop up.
In this case, we see a yellow alarm that has happened due to an alarm condition in the PBSC_W granule. To investigate further, the raised "PBSC_W" is pushed to expand the panel to look like Figure 4b. We then see in the right hand side of the panel that high voltage channel HV_PBSC_W_1_SM_5_00 has caused the yellow alarm condition. To acknowledge the alarm, the yellow square to the left of the "PBSC_W" can be pushed. To acknowledge all alarms, the yellow square in the upper left hand corner of the panel can be pushed. Although the alarm has been acknowledged, the button still retains it's color until the offending channel has been returned to it's normal operating conditions.

In the $EPICS/epicsB/baseB/src directory (Figure 1) are located the drivers that allow for communication between the IOC and the 1458 mainframe. For a detailed description of the functionality of these various drivers, the reader is directed to the theses referred to above.

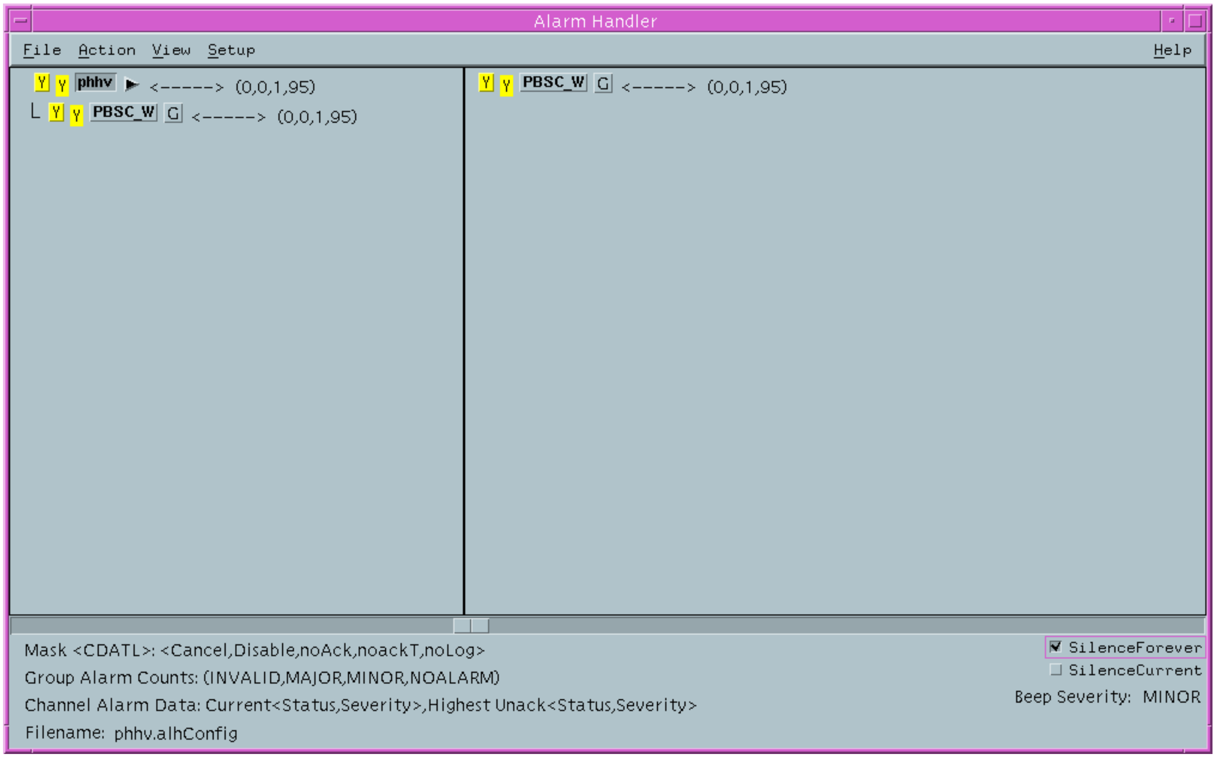

Figure 4a: EPICS Alarm Handler Panel (not expanded)

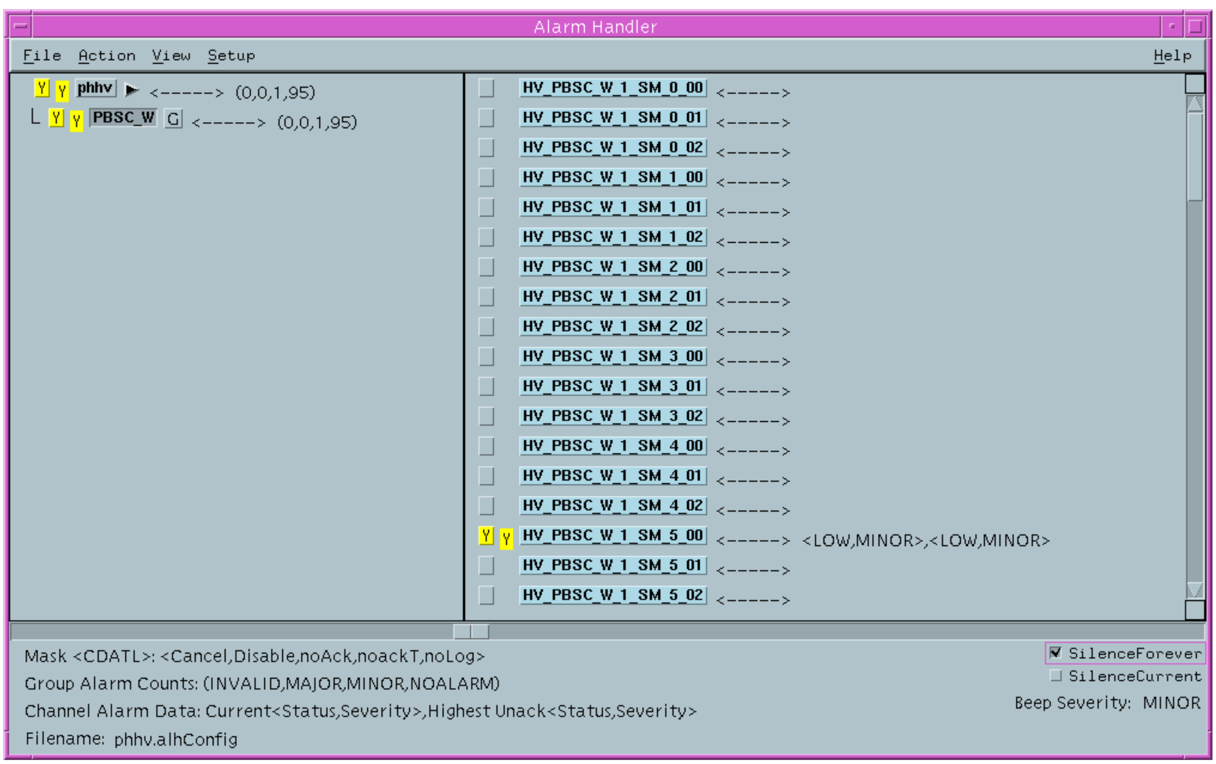

Figure 4b: EPICS Alarm Handler Panel (expanded)

In the $EPICS/epicsB/ioc directory (Figure 1) are located the bootscripts that are used to load the compiled (on phoncs0) programs onto the IOCs. These bootscripts can be pointed to in the 'startup' script listed in Table 1 or can be loaded manually by issuing the VxWorks command “< load” from the “iocondev3>” prompt. In addition, there is a bootlog file that indicates what a successful boot should look like and is useful for debugging purposed.

MEDM GUI

The High Voltage Control area on PHONCS was revised to account for the increased complexity of the PHENIX high voltage system in the year 2000. During the engineering run in 1999, only one High Voltage Arcnet network string with one VME controller was present to operate the subset of the PHENIX detector systems that existed then. The final High Voltage control network is more modular. Up to four High Voltage ArcNet networks with one VME controller each will be installed for the West_South, West_North, East_South, East_North quarters of PHENIX. Presently, two networks are controlling the West and the East Carriages.

The new Makefile and directory structure facilitates the creation of EPICS database files separately for every crate controller. This new directory structure can be seen in Figure 5. Different subsystem High Voltage configurations can be set up easily and the overall control interface (MEDM GUI) built accordingly.

This chapter describes what configuration files are required and how to use the revised Makefile.

Required Subsystem Input

==

Every subsystem needs to provide two configuration files that:

a) Describe which HV module is in which slot in which mainframe -- > e.g. dc_west_descfile (see Figure 6).

b) Describe which channel is in which HV module -- > e.g. dc_west_channel (see Figure 7).

This information is entered into the Objectivity Database (for the time being with the program >>test_stand<<). In a next step, the information on the subsystem configuration is read out again from the Objectivity Database (for the time being with program >>db_datextr<<) and written into a combined configuration file (-- > e.g. dc_w.dat) that is used for building the EPICS database file and the MEDM User Interface (see Objectivity Chapter for further information on test_stand and db_datextr).

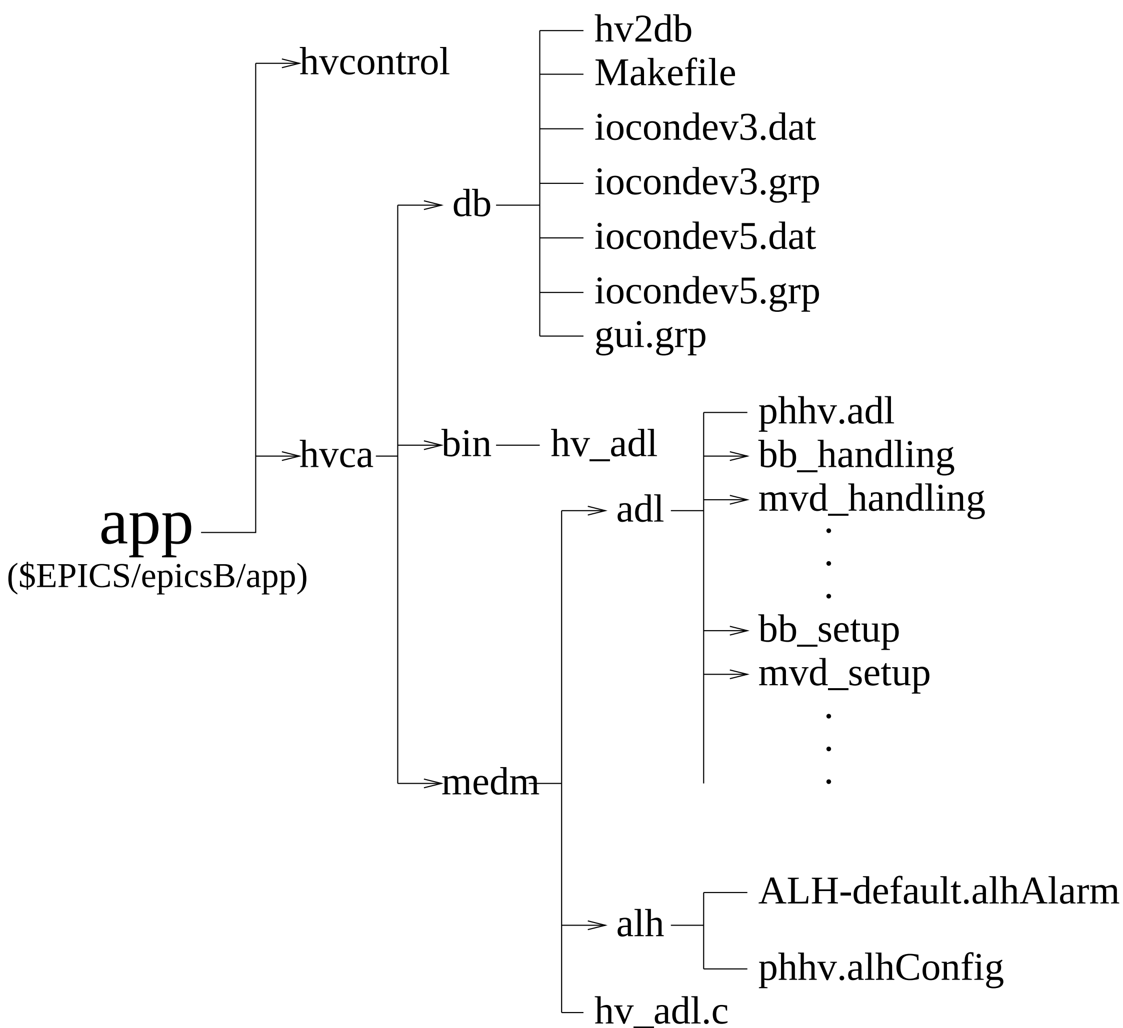

Figure 5: app (Application) directory sub-structure

```
##########################################
# DC_W HV CRATE CONFIGURATION FILE J.M.HEUSER 01/27/2000
##########################################
#
# Format:
# 1. ModName 2. ModSN 3. ModOwner 4. ModType 5. ModGranule
# 6. ModSlot 7. MFName 8. CrateCntr 9. MFID 10. MFType
# 11. MFLocation
##########################################

HV_DC_W_S_M15 B46893 DC_W 1469N DC_W 15 mf50 iocondev3 50 LeCroy1458 west_arm
HV_DC_W_S_M14 B33489 DC_W 1469N DC_W 14 mf50 iocondev3 50 LeCroy1458 west_arm
........
```

Figure 6 dc_west_descfile (Database Description File)

```
##########################################
# DC_W Channel Name file J.M.HEUSER 01/27/2000
##########################################
#
# The positions and types of modules are defined separately in the
# Database Description File!
#
# convention for channel names: SOUTH - X1,UV1 wires
# NORTH - X2,UV2 wires
# file format:
# 1. ModuleSN 2. ChannelNumber 3. ChannelName 4. MainframeID
##########################################

B28930 0 HV_DC_W_S_KS00_UV1_P 50
B28930 1 HV_DC_W_S_KS00_X1__P 50
B28930 2 HV_DC_W_S_KS01_UV1_P 50
...
```

Figure 7 dc_west_channel (Channel Description File)

1	2	3	4	5	6	7	8	9
HV_DC_W_S_KS00_X1__C	05	01	01	02	01	02	01	+0
HV_DC_W_S_KS00_X1__G	05	01	01	02	03	01	09	+0
HV_DC_W_S_KS00_X1__P	05	01	01	02	03	01	01	+0
HV_DC_W_S_KS00_X1__G	05	01	01	02	03	01	09	+0
HV_DC_W_S_KS00_X1__B	05	01	01	02	03	01	017	+0
HV_DC_W_S_KS00_UV1_C	05	01	01	02	01	02	00	+0
HV_DC_W_S_KS00_UV1_P	05	01	01	02	03	01	00	+0
HV_DC_W_S_KS00_UV1_G	05	01	01	02	03	01	08	+0
HV_DC_W_S_KS00_UV1_B	05	01	01	02	03	01	016	+0
	...							
HV_DC_W_S_KS00-03_PBS	05	01	01	02	03	00	000	+0
HV_DC_W_S_KS00-03_GBS	05	01	01	02	03	00	001	+0
HV_DC_W_S_KS00-03_BBS	05	01	01	02	03	00	002	+0
	...							

1	10	11	12	13	14	15	16	17	18
HV_DC_W_S_KS00_X1__C	3	2	25	75	+100	-4800	1	100	1471N
HV_DC_W_S_KS00_X1__G	3	2	25	75	+100	-1700	1	100	1469N
HV_DC_W_S_KS00_X1__P	3	2	25	75	+100	-2700	0	0	1469N
HV_DC_W_S_KS00_X1__G	3	2	25	75	+100	-1700	0	0	1469N
HV_DC_W_S_KS00_X1__B	3	2	25	75	+100	-1000	0	0	1469N
HV_DC_W_S_KS00_UV1_C	3	2	25	75	+100	-4800	1	100	1471N
HV_DC_W_S_KS00_UV1_P	3	2	25	75	+100	-2700	0	0	1469N
HV_DC_W_S_KS00_UV1_G	3	2	25	75	+100	-1700	0	0	1469N
HV_DC_W_S_KS00_UV1_B	3	2	25	75	+100	-1000	0	0	1469N
	...								
HV_DC_W_S_KS00-03_PBS	3	2	25	75	+100	+2700	0	0	1469N
HV_DC_W_S_KS00-03_GBS	3	2	25	75	+100	+1700	0	0	1469N
HV_DC_W_S_KS00-03_BBS	3	2	25	75	+100	+1000	0	0	1469N
	...								

Figure 8a. Format of '.dat' file for input to Makefile in db directory.
Note: Bottom three rows are 'Bulk Supply' channels

Note:

1) The drift chamber uses a drift chamber specific version of db_datextr that re-groups the channels in the required way !!
   The program is: /home/phoncs/heuser/HV/src/db_datextr
   /home/phoncs/heuser/HV/bin/SunOS.sparc.5.6

2) This *.dat file can be written also by hand (even though this is not recommended). The format shown in Figure 8.

2. Makefile Group Files

================================

One *group file* (grp) for every VME crate controller is required as well as one group file for the Graphical User Interface (GUI). The lines contain the High Voltage granule number and the granule name. For every granule listed in the group

1. channel name
2. HV group number:
3. 1-ch. enabled, 0-ch. disabled
4. arcnet ID of C.C.; always 1
5. MF arcnet ID (decimal)
6. MF slot number
7. 01-normal(1469)ch., 00-bulk supply(1469), 02-normal(other)ch.
8. channel number in module
9. default voltage setpoint
10. voltage dead zone
11. current dead zone
12. ramp up V/s
13. ramp down V/s
14. trip current uA
15. voltage limit
16. 1=ramp trip enabled, 0=r. t. disabled
17. 0=ramp trip zero
18. module type

Figure 8b. Description of .dat file (column numbers)

Table 3. Group Numbers, Granules and '.dat' Files

Granule	Grp #	.dat File	Granule	Grp #	.dat File	Granule	Grp #	.dat File
MVD	01	mvd.dat	PC_E	06	pc_e.dat	TOF_E	11	tof_e.dat
BB	02	bb.dat	PC_W	07	pc_w.dat	PBGL_E	12	pbgl_e.dat
ZDC	03	zdc.dat	TEC_E	08	tec_e.dat	PBSC_E	13	pbsc_e.dat
DC_E	04	dc_e.dat	RICH_E	09	rich_e.dat	PBSC_W	14	pbsc_w.dat
DC_W	05	pc_e.dat	RICH_W	10	rich_w.dat	-	-	-

files, a subsystem High Voltage setup file >> *.dat << has to exist. The naming conventions can be seen in Table 3.

A set of group files where the granules DC_W and PBSC_W are in operation in the west arm and the granules DC_E and TEC_E are in operation in the east arm can be seen in Figure 9.

3. EPICS Database Creation and MEDM GUI adl File Creation

==

a) build the EPICS database files for the IOCs (e.g. iocondev3):

By issuing the the command 'make -s iocondev3' at the > prompt, text similar to the following should be echoed to the screen.

iocondev3.grp	iocondev5.grp	gui.grp
5 DC_W	4 DC_E	4 DC_E
14 PBSC_W	8 TEC_E	5 DC_W
		8 TEC_E
		14 PBSC_W

Figure 9. Set of Group Files

phoncs0:db> make -s iocondev3

BUILDING THE EPICS DATABASE FILE FOR IOCONDEV3
==

OK: Group file >>iocondev3.grp<< found

OK: Intermediate group file >>phhv.grp<< created

REQUESTED SUBSYSTEMS:
=======================:

5 DC_W
14 PBSC_W

* DC_W listed in group file
–> OK: dc_w.dat file found
PBSC_W listed in group file
–> OK: pbsc_w.dat file found

Intermediate input file >>phhv.dat<< created

CREATING THE DATABASE ...

Configuration file ready: iocondev3.config
EPICS database file ready: iocondev3.db

READY !!

b) Build the GUI:
——————

By issuing the the command 'make -s gui' at the > prompt, text similar to the following should be echoed to the screen.

phoncs0:db> make -s gui

BUILDING THE MEDM HV-CONTROL GUI
======================

OK: Group file >>gui.grp<< found

OK: Intermediate group file >>phhv.grp<< created

REQUESTED SUBSYSTEMS:
4 DC_E
5 DC_W
8 TEC_E
14 PBSC_W

* DC_E listed in group file
–> OK: dc_e.dat file found
DC_W listed in group file
–> OK: dc_w.dat file found
TEC_E listed in group file
–> OK: tec_e.dat file found
PBSC_W listed in group file
–> OK: pbsc_w.dat file found

Intermediate input file >>phhv.dat<< created

BUILDING MEDM GUI ...
Using datafile phhv.dat

crateinfo OK
crateinfo OK
crateinfo OK
crateinfo OK
crateinfo OK
crateinfo OK
crateinfo OK
crateinfo OK
There is a crate# 34
There is a crate# 50
There is a crate# 51
There is a crate# 52

There is a crate# 53
There is a crate# 65
There is a crate# 81
There is a crate# 85

WORKING ON GROUP #4
Initializing variables to default values.
Creating file: phhv_DC_E_1.adl
.
.
.
.
WORKING ON GROUP #5
Initializing variables to default values.
.
.
.
.

Objectivity Database

PHENIX has chosen Objectivity [1] as the main database for the experiment. This is an Object Oriented/C++ database that will contain data for fast and straightforward access. The relevant directory structure of this area can be seen in Figure 10. At the top, we see the bin directory which includes the executable programs. After this, we see the ginc and gsrc (for *generated* include and source) directories. The Objectivity compiler generates these directories and their contents after a 'make clean' and then 'make' are executed in the src directory. Also in this figure, the ddl (data definition language) directory can be seen to contain four different ddl files. These four files define the *persistant* capable classes of the *schema*. For further information regarding the schema, the reader is referred to the Objectivity manuals located in the counting house.

After the ddl directory, we see the main include directory which contains the EPICS extension ezca (ez channel access) header. It is using this header, and then through inheirtance of member functions from the rthv_channel class (via gen_epics_device.h) that the connection between EPICS and Objectivity is made. For additional information, the reader is referred to the web page written by Chris Witzig (using doc++).
(http://www.phenix.bnl.gov/ phoncs/oncs/code_documentation/HVclasses/aindex.html).

Finally, in the src directory, are located the chan, dat, and desc directories that contain the input to and output from the test_stand and db_datextr programs as will be described below. In addition, we see the HV classes (cfhv_mainframe.cc, hv_channel.cc etc.) and the programs (db_mgr.cc, db_datextr.cc, etc.) that are derived from them.

The High Voltage Objectivity implementation will contain three different types of databases:

1) The Configuration Database. This will contain information about what type of module is contained in which slot of what mainframe. This database is expected to remain relatively constant in size, increasing slowly as different modules and/or mainframes are swapped in and out in the event of device failure. This database is created and populated by using the test_stand program as described below.

2) The Readback Database. This will contain information about the HV channels of a particular subsystem at a given time. In particular, the measured voltage and the measured current will be recorded at regular times. In distinction from the Configuration Database, the Readback database is expected to grow rapidly as more HV data is taken. It is expected that this will be backed up onto tape and then deleted periodically. This database is also created using the test_stand program but

[1] http://www.objectivity.com

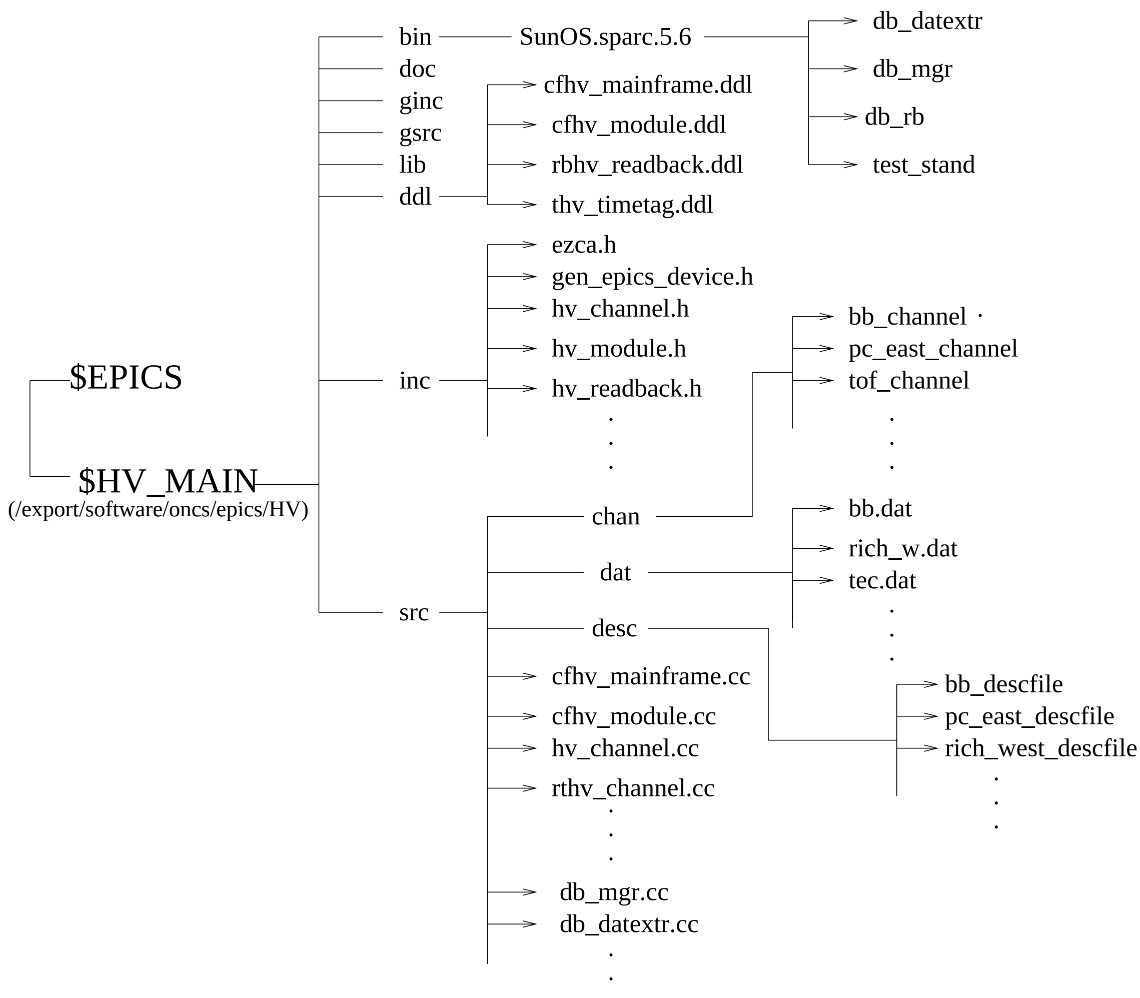

Figure 10: HV/Objectivity Directory Structure

is populated using the db_rb program as described below.

3) The Time Tag Database. This will contain the time stamp information for quick and easy access. This is also created using test_stand and populated using db_rb.

Each of the individual granules will have their corresponding Configuration, Readback and Time Tag databases.

test_stand

The test_stand program is used to create and populate various different Objectivity Databases. The source and executable program are located in the $HV_MAIN/src and $HV_MAIN/bin/SunOS.sparc.5.6/ respectively (see Figure 10). This program takes as inputs, a *Database Description File* and a *Channel Name File* located in the $HV_MAIN/src/desc and $HV_MAIN/src/chan respectively. These files contain information such as the name of the modules, the name of the channels, the slot number etc. **A first step in the operation of any of the HV subsystems that use the LeCroy mainframes is the creation of these two files!**

In addition to creating the configuration database, this program can be used to operate a subset of the detector in real time through the use of the EPICS *ezca* extension. This will not be discussed in this manual however.

db_datextr

Once the Configuration Database has been populated, the EPICS database can be derived from it. The db_datextr program is used to *extr*act the intermediate *.dat* file from the Objectivity Database from which the EPICS database is created (see MEDM GUI Chapter).

Table 4: Parameters Readback from EPICS Database Into Objectivity (Readback) Database

1.	Channel Name	4.	Current (μA)
2.	Demand Voltage (V)	5.	Status
3.	Voltage (V)		

PC_W	pc_west_config	pc_west_tag	pc_west_rb
DC_W	dc_west_config	dc_west_tag	dc_west_rb
EMC_W_B	emc_w_b_config	emc_w_b_tag	emc_w_b_rb

Figure 11: granlist File for Input to db_rb

db_rb

The HV readback data is placed into the Readback Database via the db_rb program. Currently there are five parameters from a given EPICS HV record that are recorded. These can be observed in Table 4. The readback container names are the granule name with the timestamp appended to it with underscores (no spaces).

The input to the db_rb program is the granlist file. This file contains the name of the granule along with the corresponding name of the configuration, readback and tag databases. An example of a granlist file can be seen in Figure 11. To run the program, type 'db_rb' at the prompt and then 'g' to get the readback values. To list readback values, choose the 'l' option instead of the 'g' option.

Tools

Objectivity provides a number of *tools* for various different actions to be performed on databases. For example, 'oodumpcatalog' typed at the 'phoncs0>' prompt displays the names of all of the different databases that exist. The Obectivity tools are located at /export/software/lic/objectivity/v_5.2/solaris4/bin/.

High Voltage Troubleshooting

Symptom:

In the MEDM GUI, the On/Off button and the spaces where the HV values are displayed is blank (white).

Possible Problem:

The database that is loaded onto the IOC does not contain the needed records. This can be confirmed by issuing the 'dbl' (database list) command at the 'iocondev>' prompt.

Solution:

The iocondev group file needs to be configured properly (see MEDM GUI Chapter) and the EPICS database needs to be rebuilt. The proper EPICS database then needs to be reloaded onto the IOC.

==================================

Symptom:

In the MEDM GUI, the On/Off button and the spaces where the HV values are displayed is blank (white).

Possible Problem:

The EPICS_CA_ADDR_LIST is not set properly.

Solution:

Either source the setup_epics script in the $EPICS/scripts directory or set it directly via 'setenv EPICS_CA_ADDR_LIST "130.199.98.103 130.199.98.105". Note; a list of the IP addresses of the IOCs is available at http://www.rhic.bnl.gov/phenix/project_info/electronics/maps/phenix_network.txt.

==================================

Symptom:

The MEDM GUI display is half blank and the IOC has difficulty maintaining communication.

Possible Problem:

The module number for the non-bulk supply 1469 module channels was not set properly.

Solution:

Set the module number (immediately after the mainframe slot number) in the '.dat' file for the non-bulk supply 1469 module channels to 01 (see MEDM GUI Chapter).

================================

Symptom:

The IOC continues to flash the message

interrupt: int_handler() : Reconfiguration interrupt!!
interrupt: int_handler() : Reconfiguration interrupt!!
interrupt: int_handler() : Reconfiguration interrupt!!
interrupt: int_handler() : Reconfiguration interrupt!!

Possible Problem:

AC Power to a mainframe that is controlled by the IOC has been turned off.

Solution:

Turn on AC power to the mainframe or reconfigure and reload the EPICS database so that the offending mainframe is not communicated with.

================================

Symptom:

The IOC continues to flash the message

interrupt: int_handler() : Reconfiguration interrupt!!
interrupt: int_handler() : Reconfiguration interrupt!!
interrupt: int_handler() : Reconfiguration interrupt!!
interrupt: int_handler() : Reconfiguration interrupt!!

Possible Problem:

The mainframe is no longer connected to arcnet.

Solution:

Check arcnet power and connections and make sure that the mainframe is connected. This can be confirmed using the 'hvcontrol' program (see Figure 1).

==================================

Symptom:

The IOC complains during booting that a mainframe does not contain all of the required cards.

Possible Problem

The database that is loaded does not match the module configuration of the actual mainframe.

Solution

Examine the mainframe and the corresponding '.dat' file(s) and make sure that each of the modules in the mainframe are accounted for. As long as a module is plugged into the mainframe, it *must* have a corresponding name and channel configuration in the proper '.dat' file(s). If it is desired that a module not be included in the database, it *must* be physically disconnected from the back of the mainframe.

==================================

Symptom:

After reboot, it takes the IOC a long time ($\approx$ 9 minutes 50 seconds for iocondev3 with BB, ZDC, DC_W, PBSC_W (720 EPICS records)) to sucessfully spawn the sequencers, thereby allowing HV operation.

Problem

The IOCs are loaded with a large number of records that require a signiciant amount of time to load.

Solution:

Purchase additional IOC(s) thereby reducing the load on the overstressed iocondev3 and iocondev5.

==================================

Symptom:

There are no input widgets that enable a desired (e.g. demand voltage) property to be entered.

Possible Problem

The sub-module number that appears in column seven of the .dat file is not correct. This number should be 00 for a bulksupply channel, 01 for a sub-bulksupply channel and 02 for a non-bulk supply (i.e. not a 1469) channel.

Solution:

Edit the .dat file to make sure that the module numbers are correct for the channels (see Figures 8a and b in MEDM GUI chapter).

==================================

This chapter is intended to instruct the reader how to initialize and operate the PHENIX HV system.

The steps to operate the PHENIX HV system are as follows:

1) Logging On

First logon to phoncs5 as phoncs (often phoncs is already logged onto phoncs5). Phoncs5 is a Sun SPARCstation 5 located in the southwest corner of the counting house.

Then 'source' the setup_epics script (type 'source setup_epics' at the '$phoncs5 : scripts >$' prompt) in the /export/software/oncs/epics/R3.13.0.beta11/scripts ($EPICS/scripts) directory.

2) Starting the MEDM GUI and Alarm Handler

After sourcing the setup_epics script, the start_medm and start_alarm commands should be active. To start the Motif Editor and Display Manager (MEDM) GUI, type 'start_medm'. After typing this, a screen similar to Figure 1 should appear.

To read and/or set high voltage values for various subsystems, click on the desired button under the 'Voltage/Current' column and choose the value.

To start the alarm handler, type 'start_alarm'. After typing this, a widget similar to Figure 2 should pop up. To operate the alarm handler, the reader is referred to the more detailed HV manual referred

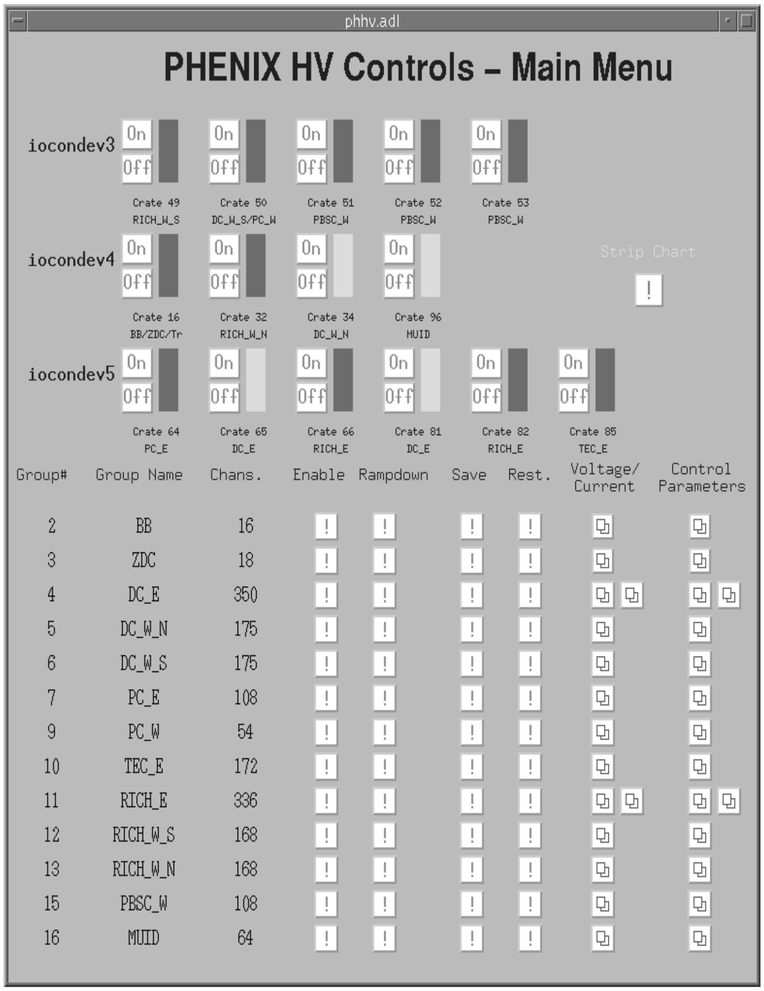

Figure 1) MEDM GUI

Figure 2) EPICS Alarm Handler

to above.

3) Rebooting the IOC

In order to run the high voltage control system, it may be necessary to reboot the Input Output Controller (IOC). The current high voltage setup has three IOCs; iocondev3, iocondev4 and iocondev5.

To logon to the IOC, type ’rsh iocondev3’ at the $' >'$ prompt on phoncs5. A ’soft’ reboot can then be attempted by typing ’control x’ at the ’iocondev3>’ prompt. If this does not work, a ’hard’ reboot should be attempted.

A ’hard’ reboot can be accomplished by pushing the red ’reset’ button on the front panel of the IOC. The IOCs are located approximately 3 meters to the west of phoncs5 (behind a wall) in the rack room of 1008 and are labeled with yellow tape.

After rebooting the IOC, the database and EPICS software can be loaded onto the IOC by typing ’< load’ at the ’iocondev3>’ prompt. After the bootscript has been executed, the cache memory assigned to the different mainframes is initialized. This can take up to ten minutes. After the cache memory is initialized, a series of numbers representing the 1458 cache memory is displayed. Finally, the sequencers that communicate between the IOC and the LeCroy 1458s are spawned. If everything goes smoothly, a note similar to Figure 3 should be displayed.

A successful bootlog of iocondev3 is located in the $IOC/bootlog

****

List of Mainframes: =

0) MF 50 Status = OK - SEQUENCER RUNNING
1) MF 34 Status = OK - SEQUENCER RUNNING

****

IOC Initialization Complete!

****

Have fun.

————————————————-

Figure 3: End of IOC Boot

directory and is called iocondev3boot. This could be helpful when attempting to debug a boot sequence.

4) Further Information

As was mentioned above, for further information the reader is referred to the more complete *PHENIX High Voltage Control System* manual (http://www.phenix.bnl.gov/ phoncs/oncs/Anc_sys/hvmanual.ps). In particular, the troubleshooting section may contain a problem, and solution, similar to a current difficulty.